

# Adaptive sparse grids for time dependent Hamilton-Jacobi-Bellman equations in stochastic control

Xavier Warin

the date of receipt and acceptance should be inserted later

**Abstract** We introduce some sparse grids interpolations used in Semi-Lagrangian schemes for linear and fully non-linear diffusion Hamilton Jacobi Bellman equations arising in stochastic control. We prove that the method introduced converges toward the viscosity solution of the problem and we show that some potentially high order schemes can be efficiently implemented. Numerical test in dimension 2 to 5 are achieved and show that deterministic methods can be used efficiently in stochastic control in moderate dimension.

## 1 Introduction

We are interested in a classical stochastic control problem whose value function is solution of the following Hamilton Jacobi equation:

$$\begin{aligned} \frac{\partial v}{\partial t}(t, x) - \inf_{a \in A} \left( \frac{1}{2} \text{tr}(\sigma_a(t, x) \sigma_a(t, x)^T D^2 v(t, x)) + b_a(t, x) Dv(t, x) \right. \\ \left. + c_a(t, x) v(t, x) + f_a(t, x) \right) = 0 \text{ in } Y \\ v(0, x) = g(x) \text{ in } \mathbf{R}^d \end{aligned} \quad (1)$$

where  $Y := [0, T] \times \mathbf{R}^d$ ,  $A$  is a complete metric space.  $\sigma_a(t, x)$  is a  $d \times q$  matrix so that  $\sigma_a(t, x) \sigma_a(t, x)^T$  is a  $d \times d$  symmetric matrix. The  $b_a$ ,  $c_a$  and  $f_a$  coefficients are functions defined on  $Y$  with values respectively in  $\mathbf{R}^d$ ,  $\mathbf{R}$  and  $\mathbf{R}$ .

Let's introduce an  $\mathbf{R}^d$ -valued controlled process  $X_s^{x,t}$  defined on a filtered

---

X. Warin  
EDF R&D & FiME, Laboratoire de Finance des Marchés de l'Energie (www.fime-lab.org)  
Tel: +33-1-47654184  
E-mail: xavier.warin@edf.fr

probability space  $(\Omega, \mathcal{F}, \mathbb{F}, \mathbb{P})$  by

$$\begin{aligned} dX_s^{x,t} &= b_a(t, X_s^{x,t})ds + \sigma_a(s, X_s^{x,t})dW_s \\ X_t^{x,t} &= x \end{aligned}$$

where  $a$  is progressively measurable. This kind of problem arises when minimizing a cost function  $J(t, x) = \mathbb{E}[\int_t^T f_a(s, X_s^{x,t})e^{c_a(s, X_s^{x,t})}ds + g(X_T^{x,t})]$  with respect to the control  $a$ . It is well known [1] that the optimal value  $\hat{J}(t, x) = \inf_a J(t, x, a)$  is a viscosity solution of equation (1).

Modified finite difference schemes can be used to treat this problem [3] approximating finite differences with non adjacent points in the stencil such that the scheme is monotone according to Barles Souganidis framework [2]. Some stochastic approaches based on the resolution of a Second Order Backward Stochastic Differential Equation have been developed in [7, 8]. They can tackle high dimension problems (an example in dimension 5 has been explored in [7]). Very recently, it has been proved that the solution of equation (1) admits a probabilistic representation by means of a Backward Stochastic Differential Equation with positive jumps [9]. In this approach, the underlying controlled process is replaced by a non controlled one and the space of controls is explored randomly by a pure jump process leading to a new algorithm based on regression [10].

In the present article, we explore the case of Semi-Lagrangian methods based on the scheme developed by Camilli Falcone [4]. Some variant have been developed by Munos Zidani [5] and Debrabant Jakobsen [6]. In this approach, the brownian motion is discretized taking two values of the order of  $\sqrt{h}$ . Starting from an initial grid discretization, the algorithm needs to interpolate the function at some points outside the grid. Recently it has been proved in [11, 12] that the monotonicity of the scheme can be somewhat relaxed leading to locally high order schemes converging to the viscosity solution of the problem. In fact, Semi-Lagrangian schemes are intrinsically monotone due to the time discretization of the problem. The interpolation only brings an error independent of the time discretization. This error can be controlled during the time step iterations if the interpolator doesn't bring too many oscillations. The idea in [11, 12] is to interpolate the solution on an  $hp$  finite element basis and truncate it when the interpolated solution oscillates too much. This approach used in a parallel framework (see [11] for details) permits to tackle problems in dimension 3 or 4 at most when the dimension of the control space is limited to one.

In order to treat moderate or high dimension problems, the classical interpolation process is too costly and the number of discretization points grows up exponentially with the dimension  $d$  of the problem, being equal to  $N^d$  where  $N$  is the number of points in one dimension. This "curse of the dimension" can be mitigated for rather smooth function by using sparse grid methods. The sparse grid method has been first introduced in [13] for partial differential equations and the idea can be traced in the Russian literature for quadrature and interpolation in [14]. Recently sparse grids have been used successfully

in many problems in high dimension (see the overview articles [15,16]). The main idea of the sparse grids consists in removing point of the full grid points that are not necessary for a good representation of regular functions. When the function is null at the boundary of a cube  $[0, 1]^d$ , and noting  $N$  the number of points in each direction of the corresponding full grid, the number of points can be reduced to  $O(N \log(N)^{d-1})$  and the error in the infinite norm only deteriorates from  $O(N^{-2})$  in the case of full grid to  $O(N^{-2} \log(N)^{d-1})$  for the sparse case with linear interpolators. Of course it can only be achieved for regular function. In fact it has been shown in [17] that the rate of convergence of sparse grids using local hat function was directly linked to the cross directives of the function interpolated. When the solution is not smooth (for example Lipschitz in our problem), there is no hope to get an accurate solution if simple sparse grids are used. In order to circumvent this problem, adaptive sparse grids have been developed in [18]. Using an estimator of the error associated to the surplus of the hierarchical representation of the solution, effective adaptive methods have been developed to get accurate estimation of the solution. It is to be noticed that this approach is mainly effective if the singularity is located in a small number of dimensions. See for example [19] to get examples of the use of adaptive sparse grids in physics. For the basket option in finance it is well known [20,21] that a change of coordinate has to be achieved such that the sparse grids works by increasing the number of points in the dimension where the singularity lies.

As for Hamilton Jacobi Bellman equation with first order derivatives, a first numerical study has been recently achieved with adaptive with sparse grids and linear hat functions [22]. In this present paper we first recall the classical regularity results associated to the problem, the time discretization scheme and time convergence results associated. In a second part, we present the sparse grids method. We prove that if the adaptation is effective, the solution calculated converges to the viscosity solution of the problem when a linear sparse interpolator is used. The rate of convergence of hat function is low (see [11] for test of convergence) and we explain how to implement some potentially locally high order schemes converging towards the viscosity solution by modifying the algorithms proposed in [23,24] and by using a truncation as explained in [11, 12]. As pointed out, on all our tests the truncation modifies only very slightly the solution. At last we numerically test the schemes developed and prove their efficiency.

## 2 Regularity results and Semi Lagrangian Scheme

For a bounded function  $w$ , we set

$$|w|_0 = \sup_{(t,x) \in Y} |w(t,x)|, \quad [w]_1 = \sup_{(s,x) \neq (t,y)} \frac{|w(s,x) - w(t,y)|}{|x - y| + |t - s|^{\frac{1}{2}}}$$

and  $|w|_1 = |w|_0 + [w]_1$ .  $C_b(Y)$  denote the space of bounded functions on  $Y$  and  $C_1(Y)$  will stand for the space of functions with a finite  $|\cdot|_1$  norm.

For  $t$  given, we denote

$$\|w(t, \cdot)\|_\infty = \sup_{x \in \mathbf{R}^d} |w(t, x)|$$

We use the classical assumption on the data of (1) for a given  $\hat{K}$ :

$$\sup_a |g|_1 + |\sigma_a|_1 + |b_a|_1 + |f_a|_1 + |c_a|_1 \leq \hat{K} \quad (2)$$

The following proposition [6] gives us the existence of a solution in the space of bounded Lipschitz functions

**Proposition 1** *If the coefficients of the equation (1) satisfy (2), there exists a unique viscosity solution of the equation (1) belonging to  $C_1(Y)$ . If  $u_1$  and  $u_2$  are respectively sub and super-solution of equation (1) satisfying  $u_1(0, \cdot) \leq u_2(0, \cdot)$  then  $u_1 \leq u_2$ .*

The equation (1) is discretized in time by the time scheme proposed by Camilli Falcone [4] for a time step  $h$ .

$$\begin{aligned} v(t+h, x) &= \inf_{a \in A} \left[ \sum_{i=1}^q \frac{1}{2q} (v(t, \phi_{a,h,i}^+(t, x)) + v(t, \phi_{a,h,i}^-(t, x))) + f_a(t, x)h \right. \\ &\quad \left. + c_a(t, x)hv(t, x) \right] \\ &:= v(t, x) + \inf_{a \in A} L_{a,h}(v)(t, x) \end{aligned} \quad (3)$$

with

$$\begin{aligned} L_{a,h}(v)(t, x) &= \sum_{i=1}^q \frac{1}{2q} (v(t, \phi_{a,h,i}^+(t, x)) + v(t, \phi_{a,h,i}^-(t, x)) - 2v(t, x)) \\ &\quad + hc_a(t, x)v(t, x) + hf_a(t, x) \\ \phi_{a,h,i}^+(t, x) &= x + b_a(t, x)h + (\sigma_a)_i(t, x)\sqrt{hq} \\ \phi_{a,h,i}^-(t, x) &= x + b_a(t, x)h - (\sigma_a)_i(t, x)\sqrt{hq} \end{aligned}$$

where  $(\sigma_a)_i$  is the  $i$ -th column of  $\sigma_a$ . We note that it is also possible to choose other types of discretization as those defined in [5].

In order to define the solution at each date, a condition on the value chosen for  $v$  between 0 and  $h$  is required. We choose a time linear interpolation once the solution has been calculated at date  $h$ :

$$v(t, x) = (1 - \frac{t}{h})g(x) + \frac{t}{h}v(h, x), \forall t \in [0, h]. \quad (4)$$

We denote  $v_h$  the discrete solution obtained for a discretization with a time step  $h$ . Following [6]

**Proposition 2** *The solution  $v_h$  of equations (3) and (4) is uniquely defined and belongs to  $C^1(Y)$ . We check that if  $h \leq (16 \sup_a \{|\sigma_a|_1^2 + |b_a|_1^2 + 1\}) \wedge 2 \sup_a |c_a|_0)^{-1}$ , there exists  $C$  such that*

$$|v - v_h|_0 \leq Ch^{\frac{1}{4}}$$

Moreover, there exists  $C$  independent of  $h$  such that

$$|v_h|_0 \leq C \quad (5)$$

$$|v_h(t, x) - v_h(t, y)| \leq C|x - y|, \forall (x, y) \in Y^2 \quad (6)$$

It is clear that if the scheme (3) is used at some discretized point  $x$  of a grid, then some interpolation are needed for the calculation of  $L_{a,h}(v)(t, x)$ . We next develop the interpolation method based on sparse grids for different function basis.

### 3 Classical Sparse Grids Interpolation

We recall some classical results on sparse grids that can be found in [16].

#### 3.1 Linear case

We first assume that the function we interpolate is null at the boundary. By a change of coordinate an hyper-cube domain can be changed to a domain  $\omega = [0, 1]^d$ . Introducing the hat function  $\phi^{(L)}(x) = \max(1 - |x|, 0)$ , we obtain the following local one dimensional hat function by translation and dilatation

$$\phi_{l,i}^{(L)}(x) = \phi^{(L)}(2^l x - i)$$

depending on the level  $l$  and the index  $i$ ,  $0 < i < 2^l$ . The grid points used for interpolation are noted  $x_{l,i} = 2^{-l}i$ . In dimension  $d$ , we introduce the basis functions

$$\phi_{\underline{l}, \underline{i}}^{(L)}(x) = \prod_{j=1}^d \phi_{l_j, i_j}^{(L)}(x_j)$$

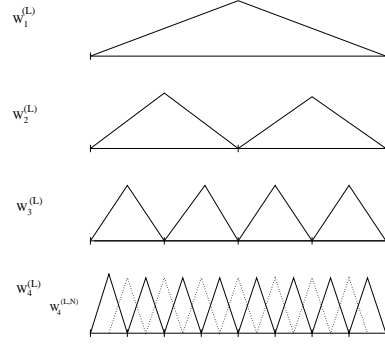
via a tensor approach for a point  $\underline{x} = (x_1, \dots, x_d)$ , a multi-level  $\underline{l} := (l_1, \dots, l_d)$  and a multi-index  $\underline{i} := (i_1, \dots, i_d)$ . The grid points used for interpolation are thus  $x_{\underline{l}, \underline{i}} := (x_{l_1, i_1}, \dots, x_{l_d, i_d})$ .

We next introduce the index set

$$B_{\underline{l}} := \{\underline{i} : 1 \leq i_j \leq 2^{l_j} - 1, i_j \text{ odd}, 1 \leq j \leq d\}$$

and the space of hierarchical basis

$$W_{\underline{l}}^{(L)} := \text{span} \left\{ \phi_{\underline{l}, \underline{i}}^{(L)}(\underline{x}) : \underline{i} \in B_{\underline{l}} \right\}$$



**Fig. 1** One dimensional  $W^{(L)}$  spaces :  $W_1^{(L)}$ ,  $W_2^{(L)}$ ,  $W_3^{(L)}$ ,  $W_4^{(L)}$  and the nodal representation  $W_4^{(L,N)}$

A representation of the space  $W_{\underline{l}}^{(L)}$  is given in dimension 1 on figure 1. The sparse grid space is defined as :

$$V_n = \bigoplus_{|\underline{l}|_1 \leq n+d-1} W_{\underline{l}}$$

*Remark 1* The conventional full grid space is defined as  $V_n^F = \bigoplus_{|\underline{l}|_\infty \leq n} W_{\underline{l}}$

At a space of hierarchical increments  $W_{\underline{l}}^{(L)}$  corresponds a space of nodal function  $W_{\underline{l}}^{(L,N)}$  such that

$$W_{\underline{l}}^{(L,N)} := \text{span} \left\{ \phi_{\underline{l}, \underline{i}}^{(L)}(\underline{x}) : \underline{i} \in B_{\underline{l}}^N \right\}$$

with

$$B_{\underline{l}}^N := \left\{ \underline{i} : 1 \leq i_j \leq 2^{l_j} - 1, 1 \leq j \leq d \right\}.$$

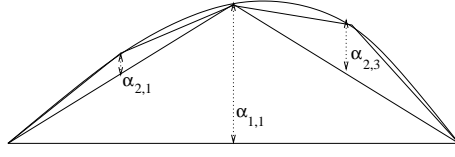
On figure 1 the one dimensional nodal base  $W_4^{(L,N)}$  is spawned by  $W_4^{(L)}$  and the dotted basis function. The space  $V_n$  can be represented as the space spawn by the  $W_{\underline{l}}^{(L,N)}$  such that  $|\underline{l}|_1 = n + d - 1$ :

$$V_n = \text{span} \left\{ \phi_{\underline{l}, \underline{i}}^{(L)}(\underline{x}) : \underline{i} \in B_{\underline{l}}^N, |\underline{l}|_1 = n + d - 1 \right\} \quad (7)$$

A function  $f$  is interpolated on the hierarchical basis as

$$I^{(L)}(f) = \sum_{|\underline{l}|_1 \leq n+d-1, \underline{i} \in B_{\underline{l}}^N} \alpha_{\underline{l}, \underline{i}}^{(L)} \phi_{\underline{l}, \underline{i}}^{(L)} \quad (8)$$

where the coefficient of the hierarchical basis  $\alpha_{\underline{l}, \underline{i}}^{(L)}$  are called the surplus (we give on figure 2 a representation of these coefficients). These surplus associated to a function  $f$  are calculated in the one dimension case for a node  $m = x_{l,i}$  as the difference of the value of the function at the node and the linear

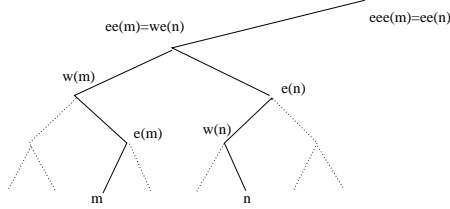


**Fig. 2** Example of hierarchical coefficients

representation of the function calculated with neighboring nodes. For example on figure 3, the hierarchical value is given by the relation :

$$\alpha^{(L)}(m) := \alpha_{l,i}^{(L)} = f(m) - 0.5(f(e(m)) + f(w(m)))$$

where  $e(m)$  is the east neighbor of  $m$  and  $w(m)$  the west one. The procedure is generalized in  $d$  dimension by successive hierarchization in all the directions. On figure 4, we give a representation of the  $W$  subspace for  $\underline{l} \leq 3$  in dimension

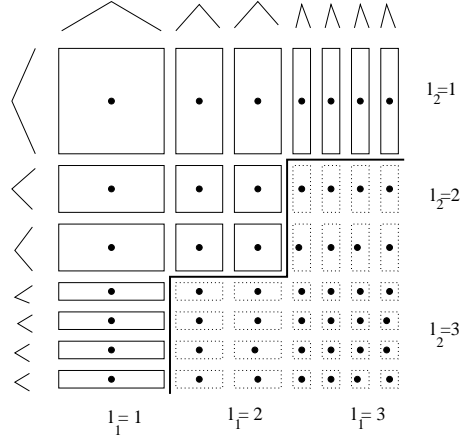


**Fig. 3** Node involved in linear, quadratic and cubic representation of a function at node  $m$  and  $n$

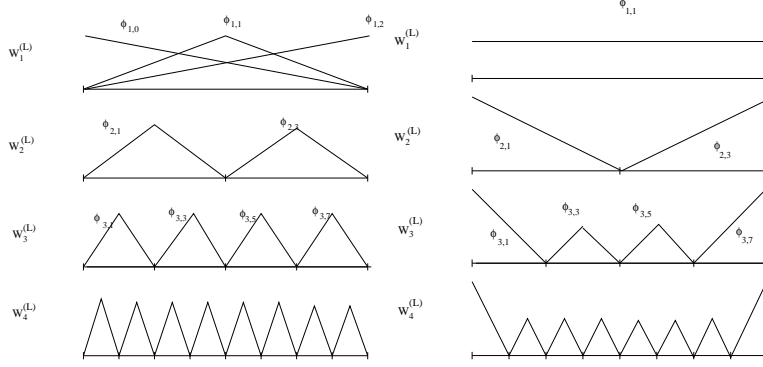
2.

In order to deal with functions not null at the boundary, two more basis are added to the first level as shown on figure 5. This approach results in many more points than the one without the boundary. As noted in [16] for  $n = 5$ , in dimension 8 you have nearly 2.8 millions points in this approximation but only 6401 inside the domain. If the boundary conditions are not important (infinite domain truncated in finance for example) the hat functions near the boundaries are modified by extrapolation (see figure 5) as explained in [16]. On level 1, we only have one degree of freedom assuming the function is constant on the domain. On all other levels, we extrapolate linearly towards the boundary the left and right basis functions, other functions remaining unchanged. So the new functions basis in 1D  $\tilde{\phi}$  becomes

$$\tilde{\phi}_{l,i}^{(L)}(x) = \begin{cases} 1 & \text{if } l = 1 \text{ and } i = 1 \\ \begin{cases} 2 - 2^l x & \text{if } x \in [0, 2^{-l+1}] \\ 0 & \text{else} \end{cases} & \text{if } l > 1 \text{ and } i = 1 \\ \begin{cases} 2^l(x - 1) + 2 & \text{if } x \in [1 - 2^{-l+1}, 1] \\ 0 & \text{else} \end{cases} & \text{if } l > 1 \text{ and } i = 2^l - 1 \\ \phi_{l,i}^{(L)}(x) & \text{otherwise} \end{cases}$$



**Fig. 4** The two dimensional subspace  $W_L^{(L)}$  up to  $l = 3$  in each dimension. The supplemental hierarchical functions corresponding to an approximation on the full grid are given in dashed lines.



**Fig. 5** One dimensional  $W^{(L)}$  spaces with linear functions with “exact ” boundary (left) and “modified ” boundary (right) :  $W_1^{(L)}$ ,  $W_2^{(L)}$ ,  $W_3^{(L)}$ ,  $W_4^{(L)}$

The interpolation error associated to the linear operator  $I^1 := I^{(L)}$  is linked to the regularity of the cross derivatives of the function [17, 23, 24]. If  $f$  is null at the boundary and admits derivatives such that  $\|\frac{\partial^{2d} u}{\partial x_1^2 \dots \partial x_d^2}\|_\infty < \infty$  then

$$\|f - I^1(f)\|_\infty = O(N^{-2} \log(N)^{d-1}) \quad (9)$$

### 3.2 High order case

As explained in [12], the observed convergence of the Semi Lagrangian method is slow. Changing the interpolator permits to get a higher rate of convergence



mainly in region where the solution is smooth. Following [23] and [24], it is possible to get higher interpolators. Using a quadratic interpolator, the reconstruction on the nodal basis gives a quadratic function on the support of the previously defined hat function and a continuous function of the whole domain. The polynomial quadratic basis is defined on  $[2^{-l}(i-1), 2^{-l}(i+1)]$  by

$$\phi_{l,i}^{(Q)}(x) = \phi^{(Q)}(2^l x - i)$$

with  $\phi^{(Q)}(x) = 1 - x^2$ .

The hierarchical surplus (coefficient on the basis) in one dimension is the difference between the value function at the node and the quadratic representation of the function using nodes available at the preceding level. With the notation of figure 3

$$\begin{aligned} \alpha(m)^{(Q)} &= f(m) - \left( \frac{3}{8}f(w(m)) + \frac{3}{4}f(e(m)) - \frac{1}{8}f(ee(m)) \right) \\ &= \alpha(m)^{(L)}(m) - \frac{1}{4}\alpha(m)^{(L)}(e(m)) \\ &= \alpha(m)^{(L)}(m) - \frac{1}{4}\alpha(m)^{(L)}(df(m)) \end{aligned}$$

where  $df(m)$  is the direct father of the node  $m$  in the tree.

Once again the quadratic surplus in dimension  $d$  is obtained by successive hierarchization in the different dimensions.

In order to take into account the boundary conditions, two linear functions  $1 - x$  and  $x$  are added at the first level (see figure 6).

A version with modified boundary conditions can be derived for example by using linear interpolation at the boundary such that

$$\tilde{\phi}_{l,i}^{(Q)}(x) = \begin{cases} \tilde{\phi}_{l,i}^{(L)} & \text{if } i = 1 \text{ or } i = 2^l - 1, \\ \phi_{l,i}^{(Q)}(x) & \text{otherwise} \end{cases}$$

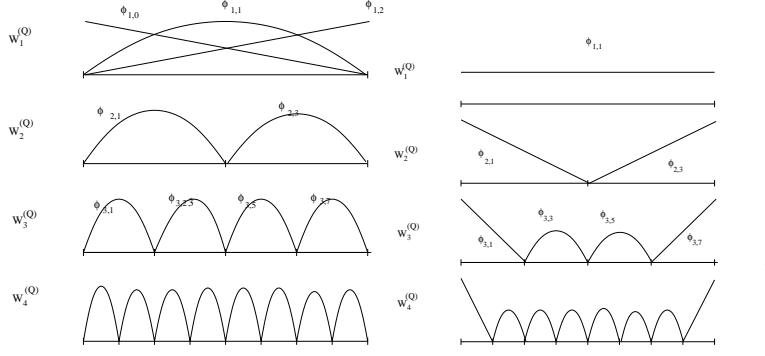
In the case of the cubic representation, on figure 3 we need 4 points to define a function basis. In order to keep the same data structure, we use a cubic function basis at node  $m$  with value 1 at this node and 0 at the node  $e(m)$ ,  $w(m)$  and  $ee(m)$  and we only keep the basis function between  $w(m)$  and  $e(m)$  [23].

Notice that there are two kinds of basis function depending of the position in the tree. The basis functions are given on  $[2^{-l+1}i, 2^{-l+1}(i+1)]$  by

$$\begin{aligned} \phi_{l,2i+1}^{(C)}(x) &= \phi^{(C),1}(2^l x - (2i+1)), \text{ if } i \text{ even} \\ &= \phi^{(C),2}(2^l x - (2i+1)), \text{ if } i \text{ odd} \end{aligned}$$

with  $\phi^{(C),1}(x) = \frac{(x^2-1)(x-3)}{3}$ ,  $\phi^{(C),2}(x) = \frac{(1-x^2)(x+3)}{3}$ .

The coefficient surplus can be defined as before as the difference between the value function at the node and the cubic representation of the function at the father node. Because of the two basis functions involved there are two kind of cubic coefficient.



**Fig. 6** One dimensional  $W^{(Q)}$  spaces with quadratic with “exact” boundary (left) and “modified” boundary (right) :  $W_1^{(Q)}$ ,  $W_2^{(Q)}$ ,  $W_3^{(Q)}$ ,  $W_4^{(Q)}$

- For a node  $m = x_{l,8i+1}$  or  $m = x_{l,8i+7}$ ,  $\alpha^{(C)}(m) = \alpha^{(C,1)}(m)$ , with

$$\alpha^{(C,1)}(m) = \alpha^{(Q)}(m) - \frac{1}{8}\alpha^{(Q)}(df(m))$$

- For a node  $m = x_{l,8i+3}$  or  $m = x_{l,8i+5}$ ,  $\alpha^{(C)}(m) = \alpha^{(C,2)}(m)$ , with

$$\alpha^{(C,2)}(m) = \alpha^{(Q)}(m) + \frac{1}{8}\alpha^{(Q)}(df(m))$$

Notice that a cubic representation is not available for  $l = 1$  so a quadratic approximation is used. As before boundary conditions are treated by adding two linear functions basis at the first level and a modified version is available. We choose the following basis functions as defined on figure 7 :

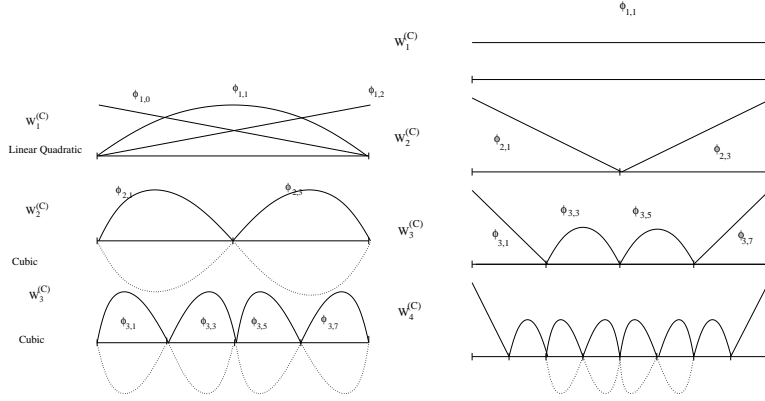
$$\tilde{\phi}_{l,i}^{(C)}(x) = \begin{cases} \tilde{\phi}_{l,i}^{(Q)} & \text{if } i \in \{1, 3, 2^l - 3, 2^l - 1\}, \\ \phi_{l,i}^{(C)}(x) & \text{otherwise} \end{cases}$$

According to [17,23,24], if the function  $f$  is null at the boundary and admits derivatives such that  $\sup_{\alpha_i \in \{2, \dots, p+1\}} \left\{ \left\| \frac{\partial^{\alpha_1 + \dots + \alpha_d} u}{\partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}} \right\|_{\infty} \right\} < \infty$  then the interpolation error can be generalized for  $I^2 := I^{(Q)}$ ,  $I^3 := I^{(C)}$  by :

$$\|f - I^p(f)\|_{\infty} = O(N^{-(p+1)} \log(N)^{d-1}), \quad p = 2, 3 \quad (10)$$

#### 4 Truncated interpolator

In order to get a higher rate of convergence where the solution is regular, we'd like to use a high order interpolator. However, the use of a high order interpolator doesn't permit to prove convergence of the calculated solution to the viscosity solution of our problem. In order to recover this convergence, we will use a truncation similar to the one developed in [11]. This approach has



**Fig. 7** One dimensional  $W^{(C)}$  spaces with cubic and “exact” boundary (left) and “modified” boundary (right) :  $W_1^{(C)}$ ,  $W_2^{(C)}$ ,  $W_3^{(C)}$ ,  $W_4^{(C)}$

to be adapted to the sparse grid case. For high order interpolator (quadratic and cubic) we truncate the interpolated value at a point  $x$  as follows : let's define for the nodal basis functions  $\phi$  :

$$K^n(x) = \{(\underline{l}, \underline{i}), \text{ such that } |\underline{l}|_1 = n + d - 1, \underline{i} \in B_{\underline{l}}^N, \text{ and } x \in \text{supp } \phi_{\underline{l}, \underline{i}}\}$$

*Remark 2* By construction, the nodal basis function  $\phi^{(L)}$ ,  $\phi^{(Q)}$ ,  $\phi^{(C)}$  have the same support.

Defining the maximum and minimum values taken by a function  $f$  at theses nodes

$$\begin{aligned} \underline{f}(x) &= \min(f(x_{\underline{l}, \underline{i}}) / (\underline{l}, \underline{i}) \in K^n(x)) \\ \bar{f}(x) &= \max(f(x_{\underline{l}, \underline{i}}) / (\underline{l}, \underline{i}) \in K^n(x)) \end{aligned} \quad (11)$$

And truncate

$$I^{p,c}(f)(x) = \underline{f}(x) \vee I^p(f)(x) \wedge \bar{f}(x)$$

*Remark 3* We have of course  $I^{1,c}(f) = I^1(f)$  but in the sequel we keep the notation  $I^{1,c}$  for genericity.

When the truncation is really achieved, the rate of the interpolation error (10) cannot be better than the one obtained by a linear interpolation given by equation (9). We hope that the truncation will only be really achieved at points where the solution is not regular.

## 5 Spatially adaptive sparse grids

When the solution is not smooth, typically Lipschitz, there is no hope to get convergence results for classical Sparse Grids (see above the interpolation error linked to the cross derivatives of the function). So classical sparse grids have to be adapted such that the solution is refined at points of irregularity. In all adaptations methods hierarchical surplus  $\alpha_{L,i}$  are used to get an estimation of the local error. These coefficients give an estimation of the smoothness of the function value at the discrete points by representing the discrete mix second derivative of the function. There is mainly two kinds of adaptation used :

- the first one is performing local adaptation and only adds points locally [27, 15, 28, 19],
- the second one is performing adaptation at the level of the hierarchical space  $W_L$  (anisotropic sparse grid). This approach detects important dimensions that needs refinement and refines all the points in this dimension [18]. This refinement is also achieved in areas where the solution can be smooth. A more local version has been developed in [26].

At the first date, the dimension adaptation developed in [18] is used to get a good approximation of the function. Because we want to implement local adaptation with refinement and coarsening, the first kind of adaptation is then used at each time step.

The algorithm used during the resolution is given in 1. Refinement is achieved

---

### Algorithm 1 Algorithm for adaptation

---

```

Use dimension adaptation at the first time step
Mesh coarsening to initialize the refinement for next step
for each time step do
  while precision and maximal level not reached do
    Refine
  end while
  Store the solution with the refined grid
  Mesh coarsening for next time step
end for
```

---

on points that have the maximal surplus above the precision required and that are located at the leave of the multidimensional tree. Each point has  $2^d$  sons. As pointed out in [22] some points constructed may not have ancestors in some dimension. In order to avoid holes in the structure missing fathers are added. At each time step, the adaptation is achieved iteratively if a maximal level of refinement is not reached. In order to prepare the next time step, the grid is coarsened by deleting points that correspond to leaves in the multidimensional tree if the associated surplus are less then 10 times the precision required. A recent description of the algorithm of refinement and mesh coarsening can be found in [22].

## 6 Discretized scheme and convergence analysis

We take the following notations :

- $P_S^N(t)$  is the set of all the points  $(\underline{L}, \underline{i})$ ,  $\underline{i} \in B_{\underline{L}}$ , at date  $t$  of the adapted Sparse Grid meshing such that in each direction the maximal level is  $N$  ( $|\underline{L}|_\infty \leq N$ )
- $P_F^N$  is the set of all points  $(\underline{L}, \underline{i})$  belonging to the corresponding full grid :

$$P_F^N = \{(\underline{L}, \underline{i}) / |\underline{L}|_\infty \leq N, \underline{i} \in B_{\underline{L}}\}$$

We note  $I_N^{p,c,Full}$  the full grid operator on  $P_F^N$  such that

$$I_N^{p,c,Full}(f) = I^{p,c}[(f(x_{\underline{L}, \underline{i}}))_{(\underline{L}, \underline{i}) \in P_F^N}],$$

$p = 1$  to  $3$  (being linear, quadratic or cubic). With similar notation

$$I_N^{p,c,S}(f) = I^{p,c}[(f(x_{\underline{L}, \underline{i}}))_{(\underline{L}, \underline{i}) \in P_S^N}].$$

As we increase the number of points during adaptation while keeping the level below  $N$ , the interpolator  $I_N^{p,c,S}$  converges towards the full grid interpolator  $I_N^{p,c,Full}$ . Using the same scheme as the one defined in [11], for each point of the adapted meshing  $x_{\underline{L}, \underline{i}} \in P_S^N(t+h)$ ,  $v_{\underline{L}, \underline{i}}(t+h)$  the estimation of the value function at the date  $t+h$  and the point  $x_{\underline{L}, \underline{i}}$  is calculated by the scheme :

$$\begin{aligned} v_{\underline{L}, \underline{i}}(t+h) &= v_{\underline{L}, \underline{i}}(t) \\ &+ \inf_{a \in A} \left[ (L_{a,h} I_N^{p,c,S}[(v_{\underline{L}, \underline{ii}}(t))_{(\underline{L}, \underline{ii}) \in P_S^N(t)}])(x_{\underline{L}, \underline{i}}) \right] \end{aligned} \quad (12)$$

With this algorithm the solution can be calculated recursively for  $t_j = jh$ ,  $j = 1, n$ . An estimation of the solution at date  $t$  is given

$$\tilde{v}(t, x) = I_N^{p,c,S}[(v_{\underline{L}, \underline{i}}(t))_{(\underline{L}, \underline{i}) \in P_S^N(t)}](x)$$

### 6.1 Convergence rate for the linear interpolator

As we increase the number of points during adaptation while keeping the level below  $N$ , the interpolator with no truncation  $I_N^{1,S}$  converges towards the full grid interpolator  $I_N^{1,Full}$ . We give some assumptions for the convergence of the adaptation

**Assumption 1** *We suppose that all surplus missing associated to points to obtain the full grid operator are below  $\epsilon$  and that  $M$  is the number of missing points*

We get the following result

**Lemma 1** *Under assumption 1, for every Lipschitz function  $f$*

$$\|I_N^{1,S}(f) - f\|_\infty \leq K2^{-N} + M\epsilon$$

*where  $M$  is the number of missing points*

*Proof* As in [19] :

$$\|I_N^{1,S}(f) - f\|_\infty \leq \|I_N^{1,S}(f) - I_N^{1,Full}(f)\|_\infty + \|I_N^{1,Full}(f) - f\|_\infty$$

$f$  function being Lipschitz, we get the classical result (see [11])  $\|I_N^{1,Full}(f) - f\| \leq K2^{-N}$ . On the other hand by assumption 1

$$\begin{aligned} \|I_N^{1,S}(f) - I_N^{1,Full}(f)\|_\infty &\leq \sum_{|\underline{l}|_\infty \leq N, x_{\underline{l}, \underline{i}} \notin P_S^N} |\alpha^L(f)_{\underline{l}, \underline{i}}| \|\phi_{\underline{l}, \underline{i}}^{(L)}\|_\infty \\ &\leq M\epsilon \end{aligned}$$

because the function basis is bounded by one.

*Remark 4* The adaptation algorithm is stopped when all the surplus calculated are below a given level  $\epsilon$ . It doesn't prove that while going on the refinement, we only get surplus below this threshold.

The next lemma proves that if missing surplus to get the full grid approximation at each time step are bounded by a given value and if the number of missing points is bounded uniformly in time, then the Sparse Grid Semi-Lagrangian approximation converges towards the viscosity solution of the problem for the linear interpolator.

**Assumption 2** *Assumption 1 is satisfied with a number  $M$  independent on the time step number .*

**Theorem 1** *Under assumption 2, using the linear interpolator, the scheme (12) satisfies for all  $j \in [0, T/n]$*

$$\|\tilde{v}(t_j, \cdot) - v(t, \cdot)\|_\infty \leq C(h^{\frac{1}{4}} + 2^{-N} + M\frac{\epsilon}{h}) \quad (13)$$

*Proof* We choose  $h \leq 1$  and satisfying the hypothesis of proposition (2). Recalling that  $v_h$  is the solution of equations (3) and (4), we directly estimate  $\tilde{v} - v_h$ . Introducing

$$e(t) = \|\tilde{v}(t, \cdot) - v_h(t, \cdot)\|_\infty$$

we estimate

$$\begin{aligned} |\tilde{v}(t, x) - v_h(t, x)| &\leq |I^1[(v_{\underline{l}, \underline{i}}(t))_{(\underline{l}, \underline{i}) \in P_S^N(t)}](x) - I^1[(v_{\underline{l}, \underline{i}}(t))_{(\underline{l}, \underline{i}) \in P_F^N}](x)| + \\ &\quad |I^1[(v_{\underline{l}, \underline{i}}(t))_{(\underline{l}, \underline{i}) \in P_F^N} - v_h(x, t)]| \\ &\leq M\epsilon + \sup_{(\underline{l}, \underline{i}) \in K^{N, FULL}(t, x)} |v_{\underline{l}, \underline{i}}(t) - v_h(t, x)| \end{aligned} \quad (14)$$

where  $K^{N, FULL}(t, x)$  defines the set of all the points  $(\underline{l}, \underline{i})$  of  $P_F^N$  corresponding to edges of the nodal cell where  $x$  lies in the full grid. Notice that

$$\forall (\underline{l}, \underline{i}) \in K^{n, FULL}(x), |x_{\underline{l}, \underline{i}} - x|_\infty < C2^{-N}. \quad (15)$$

For  $(\underline{l}, \underline{i}) \in K^{N, FULL}(t, x)$ , we introduce  $V := v_{\underline{l}, \underline{i}}(t) - v_h(t, x)$ . Using equation (12), the classical relation  $|\inf \cdot - \inf \cdot| \leq \sup |\cdot - \cdot|$ , the fact that the data in equation (1) belong to  $C_1(Y)$  and equation (15) we get

$$\begin{aligned}
|V| \leq & \frac{1}{2q} \sum_{i=1}^q \left[ \sup_a |I^1[(v_{\underline{l}, \underline{i}\underline{i}}(t-h))_{(\underline{l}, \underline{i}\underline{i}) \in P_S^N(t-h)}](\phi_{a,h,i}^+(t-h, x_{\underline{l}, \underline{i}})) \right. \\
& - v_h(t-h, \phi_{a,h,i}^+(t-h, x))| \\
& + \sup_a |I^1[(v_{\underline{l}, \underline{i}\underline{i}}(t-h))_{(\underline{l}, \underline{i}\underline{i}) \in P_S^N(t-h)}](\phi_{a,h,i}^-(t-h, x_{\underline{l}, \underline{i}})) \\
& \left. - v_h(t-h, \phi_{a,h,i}^-(t-h, x))| \right] \\
& + h \sup_a |c_a|_0 |I^1[(v_{\underline{l}, \underline{i}\underline{i}}(t-h))_{(\underline{l}, \underline{i}\underline{i}) \in P_S^N(t-h)}](x_{\underline{l}, \underline{i}}) - v_h(t-h, x_{\underline{l}, \underline{i}})| \\
& + h |v_h|_0 |c_a|_1 C 2^{-N} + h \sup_a |f_a|_1 C 2^{-N} \tag{16}
\end{aligned}$$

Similarly, we obtain :

$$\begin{aligned}
|\phi_{a,h,i}^-(t, x_{\underline{l}, \underline{i}}) - \phi_{a,h,i}^-(t, x)| & \leq C 2^{-N} (1 + \sup_a |b_a|_1 h + \sup_a |(\sigma_a)_i|_1 \sqrt{hq}) \\
& \leq C 2^{-N} (1 + C(\sqrt{hq} + h)) \\
& \leq C 2^{-N}
\end{aligned}$$

and similar result is obtained for  $\phi^+$ .

Using the fact that  $|v_h|_1$  is bounded independently on  $h$ , the previous estimation for  $\phi^+$ , one gets the estimate

$$\begin{aligned}
|I^1[(v_{\underline{l}, \underline{i}\underline{i}}(t-h))_{(\underline{l}, \underline{i}\underline{i}) \in P_S^N(t-h)}](\phi_{a,h,i}^+(t-h, x_{\underline{l}, \underline{i}})) - v_h(t-h, \phi_{a,h,i}^+(t-h, x))| & \leq \\
\|\tilde{v}(t-h, \cdot) - v_h(t-h, \cdot)\|_\infty + C |v_h|_1 2^{-N} \tag{17}
\end{aligned}$$

Using the fact that  $|f_a|_1, |c_a|_1$  are bounded independently of  $a$  as in [11], equations (16) and (17) give:

$$|V| \leq \|\tilde{v}(t-h, \cdot) - v_h(t-h, \cdot)\|_\infty (1 + h\hat{K}) + C 2^{-N} \tag{18}$$

where the constant  $C$  depends on  $\tilde{K}$ ,  $|v_h|_1$ ,  $\hat{K}$ .

So using equations (14) and (18)

$$e(t) \leq (1 + h\hat{K})e(t-h) + C(2^{-N} + M\epsilon)$$

Using the fact that  $e(0) = 0$  and using discrete Gronwall Lemma

$$e(t_j) \leq C e^{\hat{C}T} \left( \frac{2^{-N}}{h} + \frac{M\epsilon}{h} \right), \quad \forall j < \frac{T}{n}$$

Moreover by using  $\|\tilde{v}(t, \cdot) - v(t, \cdot)\|_\infty \leq \|\tilde{v}(t, \cdot) - v_h(t, \cdot)\|_\infty + |v_h - v|_0$  and the proposition (2) we get  $\forall j \in (0, T/n)$  :

$$\|\tilde{v}(t_j, \cdot) - v(t_j, \cdot)\|_\infty \leq C(h^{\frac{1}{4}} + \frac{2^{-N}}{h} + M\frac{\epsilon}{h}) \tag{19}$$

## 6.2 High order interpolator estimator

The case of high order is less obvious in the general case. We will impose that the discretized grid is refined enough for 2 schemes with solutions  $\tilde{v}^-$  and  $\tilde{v}^+$  converging towards the viscosity solution and such that  $\tilde{v}^- \leq v \leq \tilde{v}^+$ .

We introduce the interpolation operators on the sparse grid (see definition in equation (11)):

$$I_{-,N}^S(f)(x) = \underline{f}(x),$$

$$I_{+,N}^S(f)(x) = \bar{f}(x),$$

and the equivalent on the full grid  $I_-^{FULL}, I_+^{FULL}$ . Let's define  $\tilde{v}_-(t_j, \cdot), \tilde{v}_+(t_j, \cdot)$  the value function defined by

$$\tilde{v}^\pm(t+h, x) = I_{\pm,N}^S[(v_{\underline{L}, \underline{i}}^\pm(t+h))_{(\underline{L}, \underline{i}) \in P_S^N(t+h)}]$$

with

$$v_{\underline{L}, \underline{i}}^\pm(t+h, x) = v_{\underline{L}, \underline{i}}^\pm(t) + \inf_{a \in A} \left[ (L_{a,h} I^\pm[(v_{\underline{L}, \underline{ii}}(t))_{(\underline{L}, \underline{ii}) \in P_S^N(t)}])(x_{\underline{L}, \underline{i}}) \right] \quad (20)$$

We use the following assumption stating that in the previous schemes, the adaptation (used in our main scheme) is accurate enough.

**Assumption 3** *We suppose that all surplus missing of the hierarchical decomposition of the solution  $v^\pm(t)$  on the sparse grid  $P_S^N(t)$  associated to points missing to obtain the full grid operator are below  $\epsilon$  and that  $M$  the number of missing points is independent on the time step.*

**Assumption 4** *The coefficient  $c_a$  is positive.*

**Theorem 2** *Under assumptions 3 and 4, using the interpolation operators  $I_N^{p,c,S}$ ,  $p = 2, 3$ , the scheme (12) satisfies for all  $j \in [0, T/n]$*

$$\|\tilde{v}(t_j, \cdot) - v(t_j, \cdot)\|_\infty \leq C(h^{\frac{1}{4}} + \frac{2^{-N}}{h} + M \frac{\epsilon}{h}) \quad (21)$$

*Proof* By recurrence, due to the truncation applied for quadratic or cubic operator and assumption 4 we easily get that

$$\tilde{v}^-(t, x) \leq \tilde{v} \leq \tilde{v}^+ \quad (22)$$

For  $\tilde{v}^-(t, x)$  and  $\tilde{v}^+(t, x)$  we can use exactly the same procedure as in theorem 1. Because the solution is only Lipschitz, all interpolation errors are the same for constant per mesh interpolation as in the linear case. Therefore we get :

$$\|\tilde{v}^-(t_j, \cdot) - v(t_j, \cdot)\|_\infty \leq C(h^{\frac{1}{4}} + \frac{2^{-N}}{h} + M \frac{\epsilon}{h}) \quad (23)$$

and

$$\|\tilde{v}^+(t_j, \cdot) - v(t_j, \cdot)\|_\infty \leq C(h^{\frac{1}{4}} + \frac{2^{-N}}{h} + M \frac{\epsilon}{h}) \quad (24)$$

The result is straightforward using equations (22), (23) and (24).



*Remark 5* Assumption 4 is not a strict restriction : using a change of unknown,  $v(t, x) = e^{-Kt}u(t, x)$  with  $K > -|c_a|_1$ ,  $u$  satisfies assumption 4.

## 7 Numerical examples

Two data structures have been tested to store hierarchical coefficients : the most compact is the one developed in [25] using a bitset to represent a position in the multidimensional tree. A second one tested is more classical : each node is represented by a multidimensional level and a multidimensional index. The efficiency of both data structure for interpolation is very similar so we decide to give results with the second data structure.

A key point for the efficiency of the method is the reduction of the cost of the interpolation procedure. Due to the adaptation step, the structure is very irregular. A classical algorithm to interpolate is to see the sparse grid as a recombination of full grids with different size mesh. The solution is interpolated on each full grid and all contributions are gathered to reconstruct the solution. This approach in the multidimensional case is not effective due to adaptation. An effective one consists in starting from the root node in the tree, then calculate its contribution to the solution and recursively go down the tree in all the dimensions testing if the point belongs to the support of the left or the right node. In that way only basis functions whose support encompass the points are evaluated giving a contribution to the solution.

The different test cases can be divided into two kinds :

- in the first two cases, the problem needs to get a quite good approximation of the solution at the boundary. Therefore we use the Sparse Grid method with the boundary points. No adaptation is used for these very simple cases.
- in all the other cases, the boundary condition is not very important . Because we are interested in what is happening far away from the boundary, one can only use an approximated solution on the boundary or one can use the Sparse Grid using extrapolation to avoid boundary points. The last cases are financial cases that permits to test the extrapolation method.

In all the cases given, the truncation appears to be useless, costing more than 5% of the total consumption time but only modifying the solution calculated at the third digit. So the results are given without truncation.

*Remark 6* Please notice that when the modified boundary is used no estimation of the error is available.

*Remark 7* When test cases are parallelized, the parallelization is only achieved by distributing the points where the value function must be estimated. No parallelization of the hierarchization procedure and the adaptation phase (location of points to be refined, points to add to the structure, points to be removed when coarsening) is performed. When a high number of processors is used, the time used in the non parallelized part of the software can be dominating.

**Table 1** Test case 1

Level	LINEAR			QUADRATIC			CUBIC		
	Err	Rate	Time	Err	Rate	Time	Err	Rate	Time
4	0.3835	—	0	0.0497	—	0	0.0325	—	0
5	0.4429	-0.14	0	0.0096	1.64	0	0.0163	0.7	0
6	0.4301	0.03	0	0.0038	0.93	0	0.0046	1.25	0
7	0.3376	0.24	1	0.0012	1.1	1	0.0004	2.3	1
8	0.1794	0.63	3	0.0004	1.15	3			
9	0.0685	0.96	6						
10	0.0249	1.01	16						
11	0.0078	1.15	41						
12	0.0022	1.25	103						
13	0.0005	1.44	258						

## 7.1 Two dimensional test cases

We take some test cases from [11]

### 7.1.1 First test case without control

Its solution is not regular

$$u(t, x) = (1 + t) \sin\left(\frac{x_2}{2}\right) \begin{cases} \sin \frac{x_1}{2} & \text{for } -2\pi < x_1 < 0 \\ \sin \frac{x_1}{4} & \text{for } 0 < x_1 < 2\pi \end{cases}$$

with

$$\begin{aligned} f_a(t, x) &= \sin \frac{x_2}{2} \begin{cases} \sin \frac{x_1}{2} (1 + \frac{1+t}{4}) (\sin^2 x_1 + \sin^2 x_2) & \text{for } -2\pi < x_1 < 0 \\ \sin \frac{x_1}{4} (1 + \frac{1+t}{16}) (\sin^2 x_1 + 4 \sin^2 x_2) & \text{for } 0 < x_1 < 2\pi \end{cases} \\ &\quad - \sin x_1 \sin x_2 \cos \frac{x_2}{2} \begin{cases} \frac{1+t}{2} \cos \frac{x_1}{2} & \text{for } -2\pi < x_1 < 0 \\ \frac{1+t}{4} \cos \frac{x_1}{4} & \text{for } 0 < x_1 < 2\pi \end{cases} \\ c_a(t, x) &= 0, \quad b_a(t, x) = 0 \quad \sigma_a(t, x) = \sqrt{2} \begin{pmatrix} \sin x_1 \\ \sin x_2 \end{pmatrix} \end{aligned}$$

On take  $Q = (0, 1] \times [-2\pi, 2\pi]^2$ , the number of time step is equal to 400. The error  $er_n$  at a step  $n$  is given in the infinite norm on the domain. The rate of convergence given is calculated as  $\log(\frac{er_n}{er_{n-1}})$ . An error of  $4e - 4$  corresponds to the time discretization error of the scheme. The solution is not regular but quadratic and cubic approximation give far better results than linear interpolation scheme. Besides it is difficult to get a steady rate of convergence.

### 7.1.2 Control problem with a regular solution [6], [5]

The regular solution is given by

$$u(t, x_1, x_2) = \left(\frac{3}{2} - t\right) \sin x_1 \sin x_2$$

**Table 2** Test case 2

Level	LINEAR			QUADRATIC			CUBIC		
	Err	Rate	Time	Err	Rate	Time	Err	Rate	Time
4	0.4760	—	31	0.0154	—	32	0.0046	—	34
5	0.3943	0.18	103	0.0072	0.74	106	0.0363	-2.06	115
6	0.2042	0.65	319	0.0032	0.80	320	0.0007	3.89	357
7	0.0717	1.04	931	0.0104	-1.17	968	0.0005	0.47	1047
8	0.0216	1.20	2608	0.0005	2.97	2700	0.0005	—	2939
9	0.0058	1.31	7109						
10	0.0012	1.52	18575						
11	0.0003	1.58	47032						

Coefficients are given by

$$\begin{aligned}
f_a(t, x) &= \left(\frac{1}{2} - t\right) \sin x_1 \sin x_2 + \left(\frac{3}{2} - t\right) \left[ \sqrt{\cos^2 x_1 \sin^2 x_2 + \sin^2 x_1 \cos^2 x_2} \right. \\
&\quad \left. - 2 \sin(x_1 + x_2) \cos(x_1 + x_2) \cos x_1 \cos x_2 \right] \\
c_a(t, x) &= 0, \quad b_a(t, x) = a \quad \sigma_a(t, x) = \sqrt{2} \begin{pmatrix} \sin(x_1 + x_2) \\ \cos(x_1 + x_2) \end{pmatrix}, \\
A &= \{a \in \mathbf{R}^2 : a_1^2 + a_2^2 = 1\}
\end{aligned}$$

$Q = (0, 1] \times [-\pi, \pi]^2$  and the number of time steps is equal to 800, the number of control equal to 400. With linear interpolation, the convergence is slow but exhibit a steady rate. With quadratic or cubic, the error is smaller but the convergence rate is more erratic. Oscillation for values around 0.0005 are linked to the time discretization error. The error and the convergence rates are calculated as in the previous case.

## 8 Portfolio optimization

All test cases extend or modify some test cases taken from [7]. We report an application to the continuous-time portfolio optimization problem in financial mathematics. Let  $\{S_t, t \in [0, T]\}$  be an Itô process modeling the price evolution of  $n$  financial securities. The investor chooses an adapted process  $\{\theta_t, t \in [0, T]\}$  with values in  $\mathbf{R}^d$ , where  $\theta_t^i$  is the amount invested in the  $i$ -th security held at time  $t$ . In addition, the investor has access to a non-risky security (bank account) where the remaining part of his wealth is invested. The non-risky asset  $S^0$  is defined by an adapted interest rates process  $\{r_t, t \in [0, T]\}$ , i.e.  $dS_t^0 = S_t^0 r_t dt, t \in [0, 1]$ . Then, the dynamics of the wealth process is described by:

$$dX_t^\theta = \theta_t \cdot \frac{dS_t}{S_t} + (X_t^\theta - \theta_t \cdot \mathbf{1}) \frac{dS_t^0}{S_t^0} = \theta_t \cdot \frac{dS_t}{S_t} + (X_t^\theta - \theta_t \cdot \mathbf{1}) r_t dt,$$

where  $\mathbf{1} = (1, \dots, 1) \in \mathbf{R}^d$ . Let  $\mathcal{A}$  be the collection of all adapted processes  $\theta$  with values in  $\mathbf{R}^d$ , which are integrable with respect to  $S$  and such that the

process  $X^\theta$  is uniformly bounded from below. Given an absolute risk aversion coefficient  $\eta > 0$ , the portfolio optimization problem is defined by:

$$v_0 := \sup_{\theta \in \mathcal{A}} \mathbb{E} \left[ -\exp \left( -\eta X_T^\theta \right) \right]. \quad (25)$$

Under fairly general conditions, this linear stochastic control problem can be characterized as the unique viscosity solution of the corresponding HJB equation. We shall first start by a two-dimensional example where an explicit solution of the problem is available. Then, we will present some results in a three, four and five dimensional situations.

At last in dimension 3, we will work on a less regular solution supposing that the investor is short of a at-the-money call at the initial date (strike  $K$ ). Then the function to optimize is

$$v_0 := \sup_{\theta \in \mathcal{A}} \mathbb{E} \left[ -\exp \left( -\eta (X_T^\theta - (S_T - K)^+) \right) \right]. \quad (26)$$

### 8.0.3 A two dimensional problem

Let  $d = 1$ ,  $r_t = 0$  for all  $t \in [0, 1]$ , and assume that the security price process is defined by the Heston model [29]:

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{Y_t} S_t dW_t^{(1)} \\ dY_t &= k(m - Y_t)dt + c\sqrt{Y_t} \left( \rho dW_t^{(1)} + \sqrt{1 - \rho^2} dW_t^{(2)} \right), \end{aligned}$$

where  $W = (W^{(1)}, W^{(2)})$  is a Brownian motion in  $\mathbf{R}^2$ . In this context, it is easily seen that the portfolio optimization problem (25) does not depend on the state variable  $s$ . Given an initial state at the time origin  $t$  given by  $(X_t, Y_t) = (x, y)$ , the value function  $v(t, x, y)$  solves the HJB equation:

$$\begin{aligned} v(T, x, y) &= -e^{-\eta x} \text{ and } 0 = -v_t - k(m - y)v_y - \frac{1}{2}c^2 y v_{yy} \\ &\quad - \sup_{\theta \in \mathbf{R}} \left( \frac{1}{2} \theta^2 y v_{xx} + \theta (\mu v_x + \rho c y v_{xy}) \right) \\ &= -v_t - k(m - y)v_y - \frac{1}{2}c^2 y v_{yy} + \frac{(\mu v_x + \rho c y v_{xy})^2}{2y v_{xx}}. \end{aligned}$$

A quasi explicit solution of this problem was provided by Zariphopoulou [30]:

$$v(t, x, y) = -e^{-\eta x} \left\| \exp \left( -\frac{1}{2} \int_t^T \frac{\mu^2}{\tilde{Y}_s} ds \right) \right\|_{\mathbf{L}^{1-\rho^2}}$$

where the process  $\tilde{Y}$  is defined by

$$\tilde{Y}_t = y \text{ and } d\tilde{Y}_t = (k(m - \tilde{Y}_t) - \mu c \rho)dt + c\sqrt{\tilde{Y}_t} dW_t.$$

We take the same parameter as in [7] ( $\eta = 1$ ,  $\mu = 0.15$ ,  $c = 0.2$ ,  $k = 0.1$ ,  $m = 0.3$ ,  $Y_0 = m$ ,  $\rho = 0$ ). The initial portfolio value is equal to 1, the maturity

**Table 3** Test case 3 : portfolio optimization in dimension 2, no adaptation , exact boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
6	-0.3678	5	-0.3622	5	-0.3629	5
7	-0.3670	14	-0.3433	15	-0.3360	16
8	-0.3565	40	-0.3555	40	-0.3565	43
9	-0.3550	105	-0.3533	109	-0.3531	116
10	-0.3539	274	-0.3535	283	-0.3535	304
11	-0.3536	700				
12	-0.3535	1757				

**Table 4** Test case 3 : portfolio optimization in dimension 2, no adaptation , extrapolated boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
6	-0.3678	3	-0.3576	3	-0.3575	4
7	-0.3668	8	-0.3522	9	-0.3519	9
8	-0.3579	24	-0.3536	24	-0.3536	26
9	-0.3551	64	-0.3535	67	-0.3535	71
10	-0.3539	173				
11	-0.3536	451				
12	-0.3535	1145				

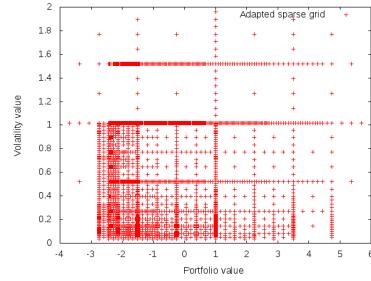
$T$  equal to 1. The reference solution is 0.3534. In all calculations, the commands  $\theta_t$  can take values in  $[-1.5, 1.5]$  and the interval is discretized with 20 steps. The resolution domain is  $[-4, 6]$  for the portfolio value and  $[0.02, 3]$  for the asset value. The number of time step is equal to 200. In table 3, we give the results obtained without adaptation for the different interpolators using an exact boundary treatment where we suppose that the boundary solution is given by a constant investment in bonds giving a boundary value equal to  $-e^{-1}$ . With this resolution parameters, the spatially converged solution is 0.3535. The rate of convergence, not stable, is not given. In table 4, the same calculation is achieved without boundary points in the approximation. Using an exact boundary treatment, quadratic and cubic converge faster than linear interpolation. Cubic is not superior to quadratic. Using extrapolated boundary, where no theoretical convergence is available, linear interpolation gives the same results as in the case of exact boundary treatment. Quadratic and cubic converges faster when no boundary conditions is imposed certainly meaning that the boundary condition imposed is not optimal. The same test case can be achieved starting with an initial level of 5 and local adaptation. Refinement is only allowed at the center of the domain  $[-2.5, 4.5] \times [0.05, 1.54]$  limiting the maximal level of refinement to 12. The use of adaptation permits to reduce the number of points used by focusing on the region of interest. It is especially effective with the quadratic or cubic interpolation.

**Table 5** Test case 3 : portfolio optimization in dimension 2, adaptation , exact boundary treatment, initial level 5

	LINEAR		QUADRATIC		CUBIC	
Precision	Solution	Time	Solution	Time	Solution	Time
0.001	-0.3593	17	-0.3495	13	-0.3535	10
0.00025	-0.3553	39	-0.3545	25	-0.3535	18
6.25e-05	-0.3542	84	-0.3536	50		
1.56e-05	-0.3537	157	-0.3535	90		

**Table 6** Test case 3 : portfolio optimization in dimension 2, adaptation , extrapolated boundary treatment, initial level 5

	LINEAR		QUADRATIC		CUBIC	
Precision	Solution	Time	Solution	Time	Solution	Time
0.001	-0.3581	10	-0.3537	4	-0.3540	3
0.00025	-0.3556	25	-0.3536	7	-0.3536	6
6.25e-05	-0.3542	54	-0.3535	16	-0.3535	13
1.56e-05	-0.3538	101				

**Fig. 8** Example of adapted meshes in dimension 2 (test case 3 with extrapolated boundary)

#### 8.0.4 A three dimensional problem

We now let  $n = 1$  , and we assume that the interest rate process is defined by the Ornstein-Uhlenbeck process:

$$dr_t = \kappa(b - r_t)dt + \zeta dW_t^{(0)}.$$

The security has the same dynamic as in the previous case. We assume that all correlations are equal to 0. The optimization problem is still given by equation (25). In this case the value function  $v(t, x, r, y)$  satisfies the HJB equation :

$$0 = -v_t - (\mathbf{L}^r + \mathbf{L}^Y)v - rxv_x - \sup_{\theta} \left\{ \theta(\mu - r)v_x + \frac{\theta^2}{2} y v_{xx} \right\}$$

where

$$\mathbf{L}^r v = \kappa(b - r)v_r + \frac{1}{2}\zeta^2 v_{rr}, \quad \mathbf{L}^Y v = k(m - y)v_y + \frac{1}{2}c^2 y v_{yy},$$

**Table 7** Test case 4 : portfolio optimization in dimension 3, no adaptation , extrapolated boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
5	-0.3458	0	-0.3387	1	-0.3403	1
6	-0.3441	4	-0.3394	3	-0.3397	3
7	-0.3437	13	-0.3379	13	-0.3378	14
8	-0.3444	46	-0.3384	48	-0.3384	51
9	-0.3396	158	-0.3383	163	-0.3383	176
10	-0.3387	517	-0.3382	543	-0.3382	570
11	-0.3384	1655	-0.3383	1670	-0.3383	1787

**Table 8** Test case 4 : portfolio optimization in dimension 3, adaptation , extrapolated boundary treatment

Precision	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
1e-3	-0.3430	15	-0.3380	10	-0.3385	9
0.00025	-0.3401	38	-0.3381	24	-0.3383	23
6.25e-05	-0.3390	90	-0.3381	53	-0.3381	54
1e-5	-0.3384	162	-0.3381	105	-0.3382	97

We take the same parameter as in the two dimensional case. As for the interest rate model we take  $b = 0.07$ ,  $r_0 = b$ ,  $\zeta = 0.3$ . The initial asset values and wealth values are equal to 1. The discretization domain is  $[-4, 10] \times [-0.2, 0.5] \times [0.02, 2]$ . The number of time steps is still equal to 200, the number of commands tested equal to 50. We use extrapolated boundary condition and results are given in table 7. Calculation are achieved with parallelization on bi processor with 24 cores. During adaptation the maximal refinement level is fixed to 11. Adaptation appears to be very effective in this case trimming the cost of calculation in all the cases.

#### 8.0.5 A four dimension example

Now take the same problem as before but modify the dynamic of the asset using a CEV-SV model (see [31] for a presentation of this model) :

$$\begin{aligned} dS_t &= \mu S_t dt + \sigma \sqrt{Y_t} S_t^\beta dW_t^{(1)}, \\ dY_t &= k(m - Y_t) dt + c \sqrt{Y_t} dW_t^{(2)} \end{aligned}$$

The optimization problem is still given by equation (25). In this case the value function  $v(t, x, r, s, y)$  satisfies the HJB equation :

$$\begin{aligned} 0 = & -v_t - (\mathbf{L}^r + \mathbf{L}^Y + \mathbf{L}^S)v - rxv_x \\ & - \sup_{\theta} \left\{ \theta(\mu - r)v_x + \theta \sigma^2 y s^{2\beta-1} v_{xs} + \frac{1}{2} \theta^2 \sigma^2 y s^{2\beta-2} \right\} \end{aligned}$$

where

$$\mathbf{L}^S v = \mu s v_s + \frac{1}{2} \sigma^2 s y v_{ss}.$$

**Table 9** Test case 5 : portfolio optimization in dimension 4, no adaptation , extrapolated boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
5	-0.3542	4	-0.3315	4	-0.3329	4
6	-0.3435	20	-0.3354	20	-0.3371	21
7	-0.3453	99	-0.3358	102	-0.3361	108
8	-0.3443	441	-0.3364	452	-0.3384	481
9	-0.3382	1832	-0.3360	1877	-0.3359	2017
10	-0.3358	7336	-0.3360	7478	-0.3360	7915
11	-0.3357	28210	-0.3352	28870	-0.3353	30143
12	-0.3358	101921	-0.3361	104950	-0.3360	108000

**Table 10** Test case 5 : portfolio optimization in dimension 4, adaptation , extrapolated boundary treatment, initial level equal to 6

Precision	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
0.001	-0.3415	610	-0.3373	302	-0.3367	289
0.00025	-0.3369	2420	-0.3349	1471	-0.3353	1458
6.25e-05	-0.3356	7703	-0.3353	5701	-0.3353	5784
1.56e-05	-0.3355	14504	-0.3350	12716	-0.3349	12860

Take  $\eta = 1$ ,  $\mu = 0.10$ ,  $\sigma = 0.3$ ,  $\beta = 0.5$  for the asset,  $k = 0.1$ ,  $m = 1$ .,  $c = 0.1$  for the diffusion process of the asset. The maturity  $T$  is still equal to 1. Parameters for the interest rate are the same as in the three dimension case. The resolution domain is  $[-5, 10] \times [-0.2, 0.5] \times [0.02, 5] \times [0.02, 5]$ . The refinement domain is defined on  $[-3.5, 8.5] \times [-0.13, 0.43] \times [0.5, 4.5] \times [0.5, 4.5]$  calculation are achieved on 24 cores. The maximal refinement level is fixed to 12. In all the cases, the convergence appears to be rather difficult and a solution with more than 3 digits is hard to find.

#### 8.0.6 A five dimensional example

We now let  $n = 2$ , and we assume that the interest rate process is defined by the Ornstein-Uhlenbeck process:

$$dr_t = \kappa(b - r_t)dt + \zeta dW_t^{(0)}.$$

While the price process of the second security is defined by an Heston model, the first security's price process is defined by a CEV-SV model :

$$\begin{aligned} dS_t^{(i)} &= \mu_i S_t^{(i)} dt + \sigma_i \sqrt{Y_t^{(i)}} S_t^{(i)\beta_i} dW_t^{(i,1)}, \quad \beta_2 = 1, \\ dY_t^{(i)} &= k_i \left( m_i - Y_t^{(i)} \right) dt + c_i \sqrt{Y_t^{(i)}} dW_t^{(i,2)} \end{aligned}$$

where  $(W^{(0)}, W^{(1,1)}, W^{(1,2)}, W^{(2,1)}, W^{(2,2)})$  is a Brownian motion in  $\mathbf{R}^5$ , and for simplicity we considered a zero-correlation between the security price process and its volatility process.



**Table 11** Sparse grid for commands : 'Level' indicates the level of the sparse grid used to approximate to value function. 'Discretization for commands' indicate the discretization used to interpolated the commands. In the case of use of sparse grids to discretize commands, the level of the sparse grid is given by 'Level C'. The optimal command is calculated on the thin grid  $64 \times 64$

Discretization for commands	Full grid $64 \times 64$		Level C = 4		Level C = 5	
Level for solution	Solution	Time	Solution	Time	Solution	Time
9	-0.2998	5295	-0.2998	89	-0.2998	173
10	-0.3083	29046	-0.3081	424	-0.3083	886

Since  $\beta_2 = 1$ , the value function of the portfolio optimization problem (25) does not depend on the  $s^{(2)}$ -variable. Given an initial state  $(X_t, r_t, S_t^{(1)}, Y_t^{(1)}, Y_t^{(2)}) = (x, r, s_1, y_1, y_2)$  at the time origin  $t$ , the value function  $v(t, x, r, s_1, y_1, y_2)$  satisfies the HJB equation:

$$0 = -v_t - (\mathbf{L}^r + \mathbf{L}^Y + \mathbf{L}^{S^1})v - rxv_x - \sup_{\theta_1, \theta_2} \left\{ \theta \cdot (\mu - r\mathbb{1})v_x + \theta_1 \sigma_1^2 y_1 s_1^{2\beta_1-1} v_{xs_1} + \frac{1}{2} (\theta_1^2 \sigma_1^2 y_1 s_1^{2\beta_1-2} + \theta_2^2 \sigma_2^2 y_2) v_{xx} \right\}$$

where

$$\mathbf{L}^r v = \kappa(b - r)v_r + \frac{1}{2}\zeta^2 v_{rr}, \quad \mathbf{L}^Y v = \sum_{i=1}^2 k_i (m_i - y_i) v_{y_i} + \frac{1}{2} c_i^2 y_i v_{y_i y_i},$$

$$\text{and } \mathbf{L}^{S^1} v = \mu_1 s_1 v_{s_1} - \frac{1}{2} \sigma_1^2 s_1 y_1 v_{s_1 s_1}.$$

We take the following parameters (as in [7]) :  $\eta = 1$ ,  $\mu_1 = 0.10$ ,  $\sigma_1 = 0.3$ ,  $\beta_1 = 0.5$  for the first asset,  $k_1 = 0.1$ ,  $m_1 = 1.$ ,  $c_1 = 0.1$  for the diffusion process of the first asset. The second asset is defined by the same parameters as in the two dimensional example:  $\mu_2 = 0.15$ ,  $c_2 = 0.2$ ,  $m = 0.3$  and  $Y_0^{(2)} = m$ . As for the interest rate model we take  $b = 0.07$ ,  $r_0 = b$ ,  $\zeta = 0.3$ . The initial asset values and wealth values are equal to 1, the maturity is equal to 1. The number of time steps is equal to 200.

The resolution domain is  $[-5, 10] \times [-0.2, 0.5] \times [0.02, 5] \times [0.015, 7] \times [0.15, 7] \times [0.04, 2.1]$ . The space of control is in dimension 2 :  $(\theta_1, \theta_2)$  are taken in  $[-1; 5, 1.5] \times [-1.5, 1.5]$ . The classical way to find the optimal control at a given point for a given date consists in discretizing the command with a thin mesh and to test all the commands to get the optimal one. We propose to use a sparse grid in dimension 2 to represent the space of commands and we interpolate the function obtained to estimate this optimal command on a very thin grid. Once the optimal command is obtained, it is used to re estimate the value function. This approach has been tested with the cubic interpolator and results are given in table 11. Results are obtained with 384 cores.

The results obtained for the valorization in dimension 5 for linear, quadratic and cubic estimator are given in table 12 with a maximal discretization level equal to 10. Calculation time are obtained for 384 cores. Results with adaptation are given in table 13.

**Table 12** Test case 6 : portfolio optimization in dimension 5, no adaptation , extrapolated boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
6	-0.7167	50	-0.2889	51	-0.2933	52
7	-0.3326	230	-0.3035	227	-0.3044	233
8	-0.2980	1032	-0.3124	1047	-0.3132	1091
9	-0.3134	4641	-0.3092	4716	-0.3091	4980
10	-0.3112	21263	-0.3089	21238	-0.3089	22500

**Table 13** Test case 6 : portfolio optimization in dimension 5 , adaptation , extrapolated boundary treatment, initial level equal to 7

Precision	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
0.001	-0.3392	1311	-0.3085	1166	-0.3091	1170
0.00025	-0.3116	2874	-0.3098	2212	-0.3097	2192
6.25e-05	-0.3092	5667	-0.3101	3817	-0.3105	3738
1.56e-05	-0.3101	10115	-0.3095	6396	-0.3095	6262

### 8.1 A less regular three dimension problem

In this problem, we have only one security defined by a CEV-SV models :

$$\begin{aligned} dS_t &= \mu S_t dt + \sigma \sqrt{Y_t} S_t^\beta dW_t^{(1)}, \\ dY_t &= k(m - Y_t) dt + c \sqrt{Y_t} dW_t^{(2)} \end{aligned}$$

The investor short of a call option wants to optimize his portfolio following equation (26). Given an initial state  $(X_t, S_t, Y_t) = (x, s, y)$  at the time origin  $t$ , the value function  $v(t, x, s, y)$  satisfies the HJB equation:

$$\begin{aligned} 0 = & -v_t - (\mathbf{L}^Y + \mathbf{L}^S)v \\ & - \sup_{\theta} \left\{ \theta \cdot \mu v_x + \theta \sigma^2 y s^{2\beta-1} v_{xs} + \frac{1}{2} (\theta^2 \sigma^2 y s^{2\beta-2}) \right\} \end{aligned}$$

where

$$\mathbf{L}^Y v = (m - y) v_y + \frac{1}{2} c^2 y v_{yy},$$

$$\text{and } \mathbf{L}^S v = \mu s v_s - \frac{1}{2} \sigma^2 s y v_{ss}.$$

The asset parameters are given by  $\mu = 0.10$ ,  $\sigma = 0.3$ ,  $\beta = 0.5$ . The diffusion asset has the parameters  $k = 0.1$ ,  $m = 1.$ ,  $c = 0.1$ . The commands  $\theta_t$  can take values in  $[-1.5, 1.5]$  and the interval is discretized with 50 steps. We solve the problem with a number of time step equal to 200. We take the solution calculated with extrapolated boundaries with the linear, quadratic and cubic interpolator. We use 24 cores for the different calculations. As it can be seen in table (14) convergence is slower than in the more regular cases. As it can be seen in table (15), the adaptation gives good results but its interest for the cubic case is not obvious.

**Table 14** Test case 7 : portfolio optimization short of a call option in dimension 3 , no adaptation , extrapolated boundary treatment

Level	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
7	0.8909	13	-0.5936	13	-0.3173	14
8	-0.3373	46	-0.4313	47	-0.4069	51
9	-0.2220	157	-0.4294	161	-0.4340	171
10	-0.4101	521	-0.4327	524	-0.4325	566
11	-0.4208	1629	0.4311	1640	-0.4309	1760
12	-0.4299	4856	-0.4310	4981	-0.4309	5325
13	-0.4303	14233	-0.4309	14527	-0.4309	15536

**Table 15** Test case 7 : portfolio optimization short of a call option in dimension 3 , adaptation with initial level equal to 9, extrapolated boundary treatment

Precision	LINEAR		QUADRATIC		CUBIC	
	Solution	Time	Solution	Time	Solution	Time
0.001	-0.3904	2452	-0.4329	1515	- 0.4308	1428
0.00025	-0.4181	3198	-0.4311	2123	- 0.4311	2002
6.25e-05	-0.4282	3910	-0.4309	2818	- 0.4309	2663
1.56e-05	-0.4294	4249				
3.9e-06	-0.4299	4541				

## References

1. W.H. FLEMING AND H.M. SONER, *Controlled Markov Processes and Viscosity Solutions*, Springer, (2005)
2. G. BARLES AND P. E. SOUGANIDIS, *Convergence of Approximation Schemes for Fully Non-linear Second Order Equation* , Asymptotic Anal., 4, ( 1991), pp. 271–283
3. J.F. BONNANS AND H. ZIDANI, *A fast algorithm for the two dimensional HJB equation of stochastic control*, ESAIM:M2AN, 38-4 , (2004), pp . 723–735
4. F. CAMILLI AND M. FALCONE , *An approximation scheme for the optimal control of diffusion processes*, Modélisation Mathématique et Analyse Numérique 29.1,(1995), pp. 97–122
5. R. MUNOS AND H. ZIDANI, *Consistency of a simple multidimensional scheme for Hamilton-Jacobi-Bellman equations*, C. R. Acad. Sci. Paris, Ser. I Math, (2005)
6. K. DEBRABANT AND E. R. JAKOBSEN., *Semi-Lagrangian schemes for linear and fully non-linear diffusion equations*, Math. Comp, no. 283 (2013), pp. 1433–1462
7. A. FAHIM, N. TOUZI. AND X. WARIN, *A Probabilistic Numerical Scheme for Fully Nonlinear PDEs*, Annals of Applied Probability 21, 4, (2011), pp. 1322–1364.
8. X. TAN , *A splitting method for fully nonlinear degenerate parabolic PDEs*, Electron. J. Probab. 18(15), (2013), pp. 1–24
9. N. LANGRENÉ, I. KHARROUBI, H. PHAM, *Discrete time approximation of fully nonlinear HJB equations via BSDEs with nonpositive jumps*, submitted
10. N. LANGRENÉ, I. KHARROUBI, H. PHAM, *A numerical algorithm for fully nonlinear HJB equations: an approach by control randomization*, submitted
11. X. WARIN, *High order methods for time dependent Hamilton-Jacobi-Bellman equations in stochastic control*, submitted
12. X. WARIN, *Some non monotone schemes for Hamilton-Jacobi-Bellman equations*, submitted
13. CH. ZWENGER, *Sparse Grids*, In Wolfgang Hackbusch, editor, *Parallel Algorithms for Partial Differential Equations*, volume 31 of *Notes on Numerical Fluid Mechanics*, (1991), pp.241-251
14. S. SMOLYAK, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Soviet Math. Dokl.,4 (1963),pp. 240-243

15. H.-J. BUNGARTZ, M. GRIEBEL, *Sparse Grids*, Acta Numerica, volume 13, (2004), pp 147-269
16. D PFLÜGER, *Spatially Adaptive Sparse Grids for High-Dimension problems*, Dissertation, für Informatik, Technische Universität München, München (2010).
17. H.-J. BUNGARTZ., *Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poisson-Gleichung*. Dissertation, Fakultät für Informatik, Technische Universität München, November 1992.
18. T. GERSTNER, M. GRIEBEL, *Dimension-Adaptive Tensor-Product Quadrature*, Computing 71, (2003) 89-114.
19. X. MA, N. ZABARAS, *An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations*, Journal of Computational Physics, 228, (2009), pp 3084-3113
20. C. REISINGER, *Numerische Methoden für hochdimensionale parabolische Gleichungen am Beispiel von Optionspreisaufgaben*, Ph.D. Thesis, Naturwissenschaftlich-Mathematischen Gesamtfakultät der Ruprecht-Karls-Universität Heidelberg, (2004).
21. C. C. W. LEENTVAAR, C. W. OOSTERLEE, *On coordinate transformation and grid stretching for sparse grid pricing of basket options*, Journal of Computational and Applied Mathematics, volume 222, issue 1, (2008)
22. O. BOKANOWSKI, J. GARKE, M. GRIEBEL, I. KLOMPMAKER, *An Adaptive Sparse Grid Semi-Lagrangian Scheme for First Order Hamilton-Jacobi Bellman Equations*, Journal of Scientific Computing, Vol 55 (3), (2013), pp. 575-605
23. H.-J. BUNGARTZ., *Concepts for higher order finite elements on sparse grids*, Proceedings of the 3.Int. Conf. on Spectral and High Order Methods, pp. 159-170, (1996)
24. H.-J. BUNGARTZ., *A Multigrid Algorithm For Higher Order Finite Elements On Sparse Grids*, ETNA. Electronic Transactions on Numerical Analysis, (1997)
25. S. DIRNSTORFER, *Numerical quadrature on sparse grids*, Technische Universität München Fakultät für Informatik, (2000)
26. J. JAKEMAN, S.G. ROBERTS, *Local and Dimension Adaptive Sparse Grid Interpolation and Quadrature*, Sparse Grids and Applications, Springer, J. Garcke and M. Griebel Editors, Springer, (2013)
27. M. GRIEBEL, *Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences*, Computing, 61(2):151-179, (1998)
28. M. GRIEBEL, *Sparse grids and related approximation schemes for higher dimensional problems*, In L. Pardo, A. Pinkus, E. Suli, and M. Todd, editors, Foundations of Computational Mathematics (FoCM05), Santander, pages 106-161. Cambridge University Press, (2006)
29. S. L. HESTON, S. L.W., *A Closed-Form Solution for Options with Stochastic Volatility with Applications to Bond and Currency Options*, The Review of Financial Studies, Vol. 6 2 327-343, (1993)
30. T. ZARIPHOUPOULOU, *A solution approach to valuation with unhedgeable risks*, Finance and Stochastics, 5 61-82., (2001)
31. R. LORD, R. KOEKKOEK, D. VAN DIJK, *A comparison of biased simulation schemes for stochastic volatility models*, Quantitative Finance, Vol. 10, 2 177-194, (2005)